APPROXIMATION OF QUADRATIC FUNCTIONS BY CIRCULAR ARCS: APPLICATION TO NUMERICALLY CONTROLLED MACHINING

A THESIS

Presented to

The Faculty of the Division of Graduate

Studies and Research

by

Henry Pritchett Cotten

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

Georgia Institute of Technology

March, 1974

APPROXIMATION OF QUADRATIC FUNCTIONS BY CIRCULAR ARCS:

APPLICATION TO NUMERICALLY CONTROLLED MACHINING

Approved:

H. L. Johnson, Chairman

S. L. Dickerson

H. M. Wadsworth

Date Approved by Chairman 2/19/74

ACKNOWLEDGEMENTS

I wish to acknowledge the cooperation of the many individuals that suggested and encouraged this research.

Thanks are due the Georgia Tech Engineering Experiment Station, especially Mr. W. A. Novak who did some initial work on the problem and Mr. G. W. Ewell who made many helpful suggestions in the preparation of the thesis.

I would like to also thank Dr. H. L. Johnson for serving as thesis advisor and thesis committee chairman, Dr. S. L. Dickerson, and Dr. H. M. Wadsworth for their criticisms and many comments.

Finally, I would like to thank my wife, Marcia, for her patience and understanding during the preparation and writing of this thesis.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

SUMMARY

The purpose to this research was to simplify the manner in which circular arc approximation could be applied to certain quadratic functions. The application of this type of approximation is of a practical interest in numerical control machining. Some of these machines use circular interpolation to simplify the number of input commands; only the beginning, end, and center coordinates of the arc are needed. The machine automatically determines the intermediate positions and moves the workpiece in a continuous arc from the beginning to the end.

Existing methods of approximation using either least squares or Chebyshev methods were not satisfactory because of either non-linear terms or excessive computations associated with circular arcs. A drawback of existing methods was the inability to fit the circular arc to a given curve within a specified error. This error was not known until after several computations were made.

Three specific quadratic functions (parabola, ellipse, and hyperbola) were investigated to determine the number of circular arcs needed for a given error. By specifying certain parameters, such as focal length and eccentricity, computer programs were used to plot normalized relationships of these parameters. This original graphical method eliminates the iterations and simultaneous solutions of previous methods and enables these design curves to be used in numerical control and other fields.

CHAPTER I

INTRODUCTION

In the production of accurate templates and other precision two-dimensional parts, numerical control machines are sometimes used. While these machines are versatile, their movements are generally limited to the use of either linear or circular line segments. If a curve is to be machined, the available straight lines or circular arcs must then be used to approximate the curve.

The approximation of complex curves by linear or circular segments is found in many applications. Lead-susceptibility curves in the gasoline industry have been approximated by linear segments [1]. In graphical work where computer displays are generated on a cathode-ray tube, line segments are used to describe curves, such as parabolas, ellipses, and hyperbolas [2]. Circular arcs have been substituted for theoretical cutter profiles in the manufacture of involute broaches, splines and hobs [3]. Templates for the production of parabolic reflectors in the microwave field have been produced at the Georgia Tech Engineering Experiment Station by the approximation of the required parabola with circular arcs.

An important consideration in this type of approximation is the minimization of the number of straight lines or circular arcs that are used. While straight segments might seem to be easier to apply in the approximation of a curve, the difficulty arises in the number of small

lines that must be used. Even after these segments are defined, there is the problem of correcting for the cutting tool radius. The spindle path must be moved parallel to the linear segment, and a new set of calculations made to determine the cutter coordinates. If circular arcs are used to approximate the curve, cutter offset is easily accomodated by a change of radius of the circular arc.

The basic problem which is treated by this thesis is the approximation of certain functions by use of circular arcs. The primary objective of this research is to arrive at efficient procedures which will permit the approximation of certain classes of functions to a desired accuracy using the circular arc interpolating capability of numerically controlled machine tools.

In order to understand the principles of numerical control machining and approximation techniques, they will be reviewed in the following sections.

Numerical Control Machining

Background [4,5]

The background of numerical control machining goes back as far as 1801 when Joseph Marie Jacquard used a perforated card system to control the motions of a loom. This idea was carried over into other fields. Some player pianos operated on similar principles and used a punched paper tape to control the action of the keys. For the next 150 years, these were the only applications of this method.

In the 1940's, several companies began to experiment with automatic methods of machining. The Bendix Corporation had the problem of

machining a complex three-dimensional cam for an injection pump. To produce a large number of inspection templates, the Parsons Corporation of Traverse City, Michigan, realized that some method of automatic machining must be used. Both companies solved the problem in similar ways. The cutting tool coordinates were calculated by using a computer. These data points were manually programmed into the milling machines. This manual method paved the way for the development of automatic machine tool controls.

The Massachusetts Institute of Technology's Servomechanisms Laboratory and the Parsons Corporation were awarded a $250,000 contract by the United States Air Force for further development of numerically controlled machine tool techniques. By 1952, this contract had resulted in the development of a prototype computer and tape controlled milling machine. This method was so successful that the Air Force appropriated $70 million in 1957 to develop continuous-path machines for the aircraft industry.

Since the 1960's and on into the 1970's, there have been a large number of advances in the numerical control industry. The price of a numerically controlled machine has dropped so that small machine shops can afford them. The number of controlled axes has grown from two or three to as many as five or six. The numerical control techniques are now being used in metal forming, drawing, wiring and spark erosion machines [6,7].

Fundamental Principles [6,8]

There are many facets to the numerical control of various machines. Reflecting the type of tool motions, these machines can be divided into

two categories: point-to-point and contouring.

The positioning, or point-to-point system is the simplest type of numerical control machine. Machines using this type of control can move the tool to specific locations, but cannot control the path by which the tool arrives at these locations. The path of the tool relative to the work is of no importance, as long as accurate positioning is the end result. This type of system cannot describe a curve since there is no coordination between the axes of the machine.

Multi-axis contouring machines comprise the second major category of numerical control machines. Contouring systems usually use servo-motors synchronized with one another. These systems can drive the machine through space along any specified path. This ability to synchronize the axes is referred to as interpolation. It is this feature of numerical control machines that is of interest in this research.

The simplest type of interpolation is linear interpolation which produces a straight line. By specifying the beginning and end coordinates, the machine will move along the straight line connecting the two points. Stacking small straight line segments together, a general-purpose contouring system can build up complex curves. The resolution of these numerical control machines is on the order of 0.0001 inch; thus, these linearly interpolated curves are, for all practical purposes, true arcs.

Since straight lines and circles are the most frequent functions found on engineering drawings, circular interpolation was added to the numerical control hardware. These two curves, linear and circular, are able to accommodate a tool radius correction by simple modification; straight lines are moved parallel to themselves, while circles only

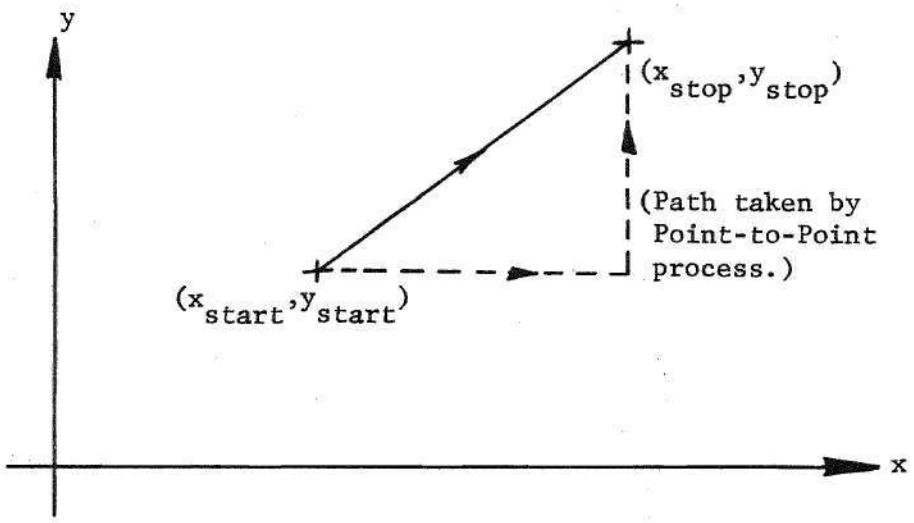
change their radii. With circular interpolation, the arc is specified in simple terms. All that is required is to specify the beginning, end, and center coordinates. The numerical control controller goes through the mathematical calculations required to move the servomotors and generate the required arc. These two types of interpolation, linear and circular, are exhibited in Figure 1.

These interpolating features are usually hard-wired into the machine's controller and are called into use by punched paper tape. To produce other curves with this type of machine would require additional circuitry in the controller. The inability to produce other types of curves prompted this research.

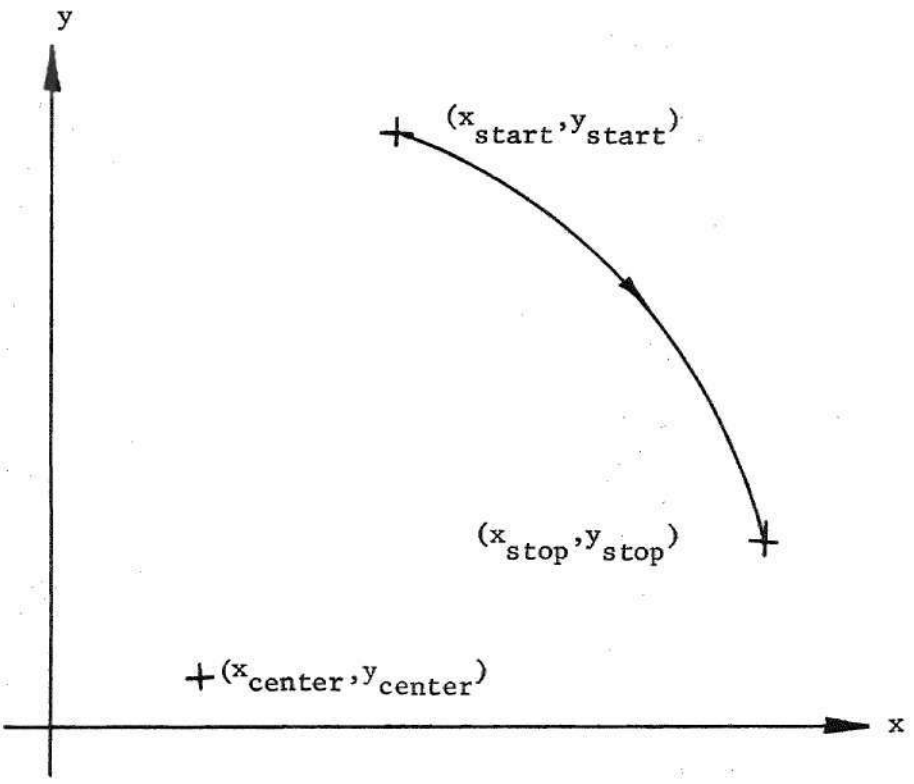
Further Numerical Control Developments

For the purpose of this research, only numerical control machines that have linear and circular interpolation are being considered. This machine along with the point-to-point machines are used in many smaller companies and machine shops. However, numerical control has advanced much further than the two interpolation features.

Numerical control machines are being directly controlled by computers instead of using the intermediate step of paper tape. By storing a number of programs in its memory and calling each one out automatically, this computer numerical control (CNC) can describe complex curves [9]. The top of the line of this type of process is direct numerical control (DNC). With DNC, the individual machines are tied to a mini-computer; the mini-computers are then supervised by a larger computer on a multiplex basis. A completely automated facility is then possible.



a. Linear



b. Circular (Clockwise)

Figure 1. Linear and Circular Interpolation

Another area of development in numerical control is the development of computer programs that would allow a draftsman, technician, or engineer to produce numerical control tapes from drawings. Like FORTRAN or ALGOL, these programs are computer languages that allow the user merely to make statements about the geometry of the part. The required spindle path to produce the shape is calculated by a computer. The designer does not have to learn the entire programming for the numerical control machine.

These languages have been expanded since the first one was developed. Illinois Institute of Technology Research Institute perfected APT, the oldest, best known, and most widely used of these languages [10,11]. Others that have been generated are APTLFT, FMILL, DAC, and SSX [12]. These latter languages are used by the designer to program sculptured surfaces in three dimensions.

While these languages are extremely powerful in their ability to define surfaces, they require a large amount of equipment and money to implement in a small company. This research is intended to help eliminate the need for large computing centers by providing design curves for certain classes of functions.

Approximation Techniques

The approximation of a function by either a set of data points or by another function is well known. Indeed, whole areas of analysis have grown from investigation of this concept.

Given a function, $f(x)$, to approximate, there is a limited number of steps to be made. The first step is to choose the approximating

function and the norm, or distance function. Other steps include the existence and uniqueness of the solution. Finally, the last step is the actual computation of the solution.

There are only a small set of functions that are generally used to approximate functions: polynomials, rational functions, and trigonometric functions. Some special functions that are sometimes used are logarithm functions, exponential functions, Bessel functions, and piecewise polynomials. However, the approximating function for this research has already been chosen to be a circular arc.

Most approximation techniques are based on the use of certain types of norms, or distance functions. The generalized distance between two functions $f(x)$ and $g(x)$ that has been widely used is [13]

$$||f(x) - g(x)|| = \left\{ \int [f(x) - g(x)]^2 \, dx \right\}^{\frac{1}{2}} \qquad (1)$$

This norm (1) is the basis for the least mean squares method of approximation. The method takes the given function, $f(x)$, and the approximating function, $g(x)$, and makes the value of the integral E

$$E = \int [f(x) - g(x)]^2 \, dx, \qquad (2)$$

a measure of the error, as small as possible. Note that E is the square of the generalized distance norm (1).

For linear approximating functions, the approximation problem can be completely solved and an explicit expression for the error obtained [14]. This type of approximation always exists and the solution is unique.

Until high-speed computers were available, it was virtually impossible to obtain best approximations in any other type of norm. There is a vast amount of literature associated with the least mean squares method, such as Hilbert spaces and Fourier series.

Another important and widely used norm is the Chebyshev norm. For continuous functions, this can be stated as

$$||f(x) - g(x)|| = \min \max |f(x) - g(x)| \qquad (3)$$

Thus a best Chebyshev approximation to f(x) minimizes the maximum value of $|f(x) - g(x)|$. A particular characteristic of the Chebyshev error curve, f(x) - g(x), is that for a set of n points, the curve alternates n-1 times and has n-2 zeros on the interval.

While these methods have been known and used for some time, approximation of functions by either linear or circular segments does not seem to have received much attention until the late 1950's. In his paper, Henry Stone presents the problem of using broken line segments to produce a best fit to a curve [1]. A general curve, y(x), is approximated by the more simple curve $y^* = a + bx$, a straight line. The original curve is broken into a number of segments, N. An optimal $y^*(x)$ is determined by the traditional techniques of calculus. Following the least squares formulation, Stone obtained the values of the variables, a and b, which minimize

$$F = \int [y(x) - a - bx]^2 \, dx \qquad (4)$$

The method that he used is the normal equation technique of finding the point at which the partial derivatives

$$\frac{\partial F}{\partial a} \text{ and } \frac{\partial F}{\partial b} \tag{5}$$

are zero. The solution of these equations for variables a and b, leads to the solving of 3N - 1 equations for 3N - 1 variables. Stone used FORTRAN and an IBM 704 computer to solve his problem.

Richard Bellman uses a technique called dynamic programming to solve Stone's problem of using linear segments [15]. The solution is much simpler since the use of dynamic programming enables him to solve for the 3N - 1 quantities in groups of three rather than simultaneously. Other approximating functions can be used instead of a striaght line. Bellman [16], Chang [17] and Nicoletti and Mariani [18] use the least squares formulation, dynamic programming, and other techniques to approximate the gain curves in control systems with linear and polynomial segments.

While there seems to be an almost endless amount of literature pertaining to the use of linear and polynomial functions, very little has been written about the use of circular arcs. This is because of the great flexibility of polynomial interpolation which says that a straight line can be passed through two points, a parabola through three, a cubic through four, and so on.

In this research, the approximating function had already been determined, i.e. circular arcs. This function, g(x), in a rectangular coordinate system is

$$g(x) = \sqrt{R^2 - (x - a_o)^2} + b_o \tag{6}$$

where a_o and b_o are the coordinates of the center of the circle and R is the radius.

If y(x) is the function to approximate, the deviation of the approximating curve g(x) from f(x) can be expressed as follows

$$f(x) - g(x) = f(x) - \sqrt{R^2 - (x - a_o)^2} - b_o \qquad (7)$$

The parameters under the square root sign, the radius R and the circle's center x-coordinate a_o, enter into the equation non-linearly, and thus present a difficult problem to solve.

S.S. Levin in his paper [3] recognized this problem. To linearize equation (7), he used the technique of weight approximations. The weighted function, q, chosen for the given function f(x) is:

$$q = f(x) + \sqrt{R^2 - (x - a_o)^2} - b_o \qquad (8)$$

where a_o, b_o, and R are the same variables as defined for equation (7). If both factors of expression (7) are multiplied by expression (8) the following is obtained

$$\Delta q = [f(x)]^2 - 2b_o f(x) + b_o^2 - R^2 + x^2 - 2a_o x + a_o^2 \qquad (9)$$

where Δq is the weighted deviation.

By introducing the designations:

$$p_o = 2b_o,\ p_1 = 2a_o,\ p_2 = R^2 - a_o^2 - b_o^2,$$

and

$$Q(x) = [f(x)]^2 + x^2 \qquad (10)$$

and substituting into expression (9), he obtains the following

$$\Delta q = Q(x) - p_o y(x) - p_1 x - p_2 \qquad (11)$$

The new parameters, p_o, p_1, and p_2 enter linearly into expression (11).

Since a Chebyshev solution minimizes the deviation between data points, the unknown parameters p_o, p_1, and p_2 can be determined from the condition of equating the weighted deviations to the points of limited deviation of Chebyshev's polynomial. Levin proceeded to pass the arc through the original function so that there would be four maxima: two on the ends of the substituted curve and two intermediate. The abscissae of the four points were determined by using the Chebyshev polynomial. This lead to the solution of four simultaneous equations using expression (11) evaluated at the four points. Once p_o, p_1, and p_2 are found, the size and position of the arc are known.

The solution, while straight forward, has some limitations. The size of the segment is strictly a guess. It is not known how close the approximation is to the original function until the equations are solved. If the error is too great, new points must be chosen to reduce the size of the segment. Once this is done; the process is repeated. This can lead to a large number of calculations to reduce the error to, say, 0.001 inch deviation.

Remez and Levin, in another paper [19], followed the same formulation as before. Instead of the simultaneous solution of four equations, Remez used a technique that he calls successive Chebyshev interpolation to determine the parameters p_o, p_1, and p_2. Here again the error is not known until after the iterative method is applied.

The typical methods of approximation are difficult to apply when the approximation function is a circular arc. It is possible to linearize the error curve by using a selective weight. However, this method and solution by Chebyshev approximation leads to the solution of simultaneous equations or iterations for each segment. In either case, the maximum deviation is not known until after the calculations are performed. What is needed is a straightforward method of determining the number of partitions from a given error size between the curves. In effect, this solves the previous problems backwards.

CHAPTER II

CIRCULAR ARC APPROXIMATION (CAR)

The purpose of this research was to simplify the manner in which circular arc approximation (CAR) may be applied to certain quadratic functions. An important consideration of the research was to produce design curves so that, by knowing a required tolerance for a machining process, the number of circular arc segments needed to approximate the function would be known.

In Chapter I, there were two papers [3,19] that contained solutions to the problem of substituting circular arcs for a function. These solutions were lengthy in their calculations; either simultaneous solution of equations or iterations had to be solved before the circular arc and actual error between the curves were known. These solutions are not readily applicable to a general class of functions, such as parabolas, ellipses, etc. Each time a new function is encountered, the same process must be repeated. If the number and size of circular arcs needed to approximate the function are known, the process of CAR could be reduced to simpler computational methods.

Because a circular arc is somewhat restrictive, more than one arc is usually required to approximate a function. To eliminate the discontinuity between arcs, the end point of one segment should be the beginning point of the next. It should be noted that there will be discontinuities in slope between the arcs since the slope of one segment

at its end will not match the slope of the next arc. This difficulty can be eliminated; one way is to use as large an arc segment as possible for a given tolerance.

The CAR technique of this research differs from the past solutions by Levin and Remez [3,19] in several respects. In this research, the circle is defined by using three points from the original function spaced equally in the x direction. Using three points to define the circle, the error at these points is zero. By careful choice of the three points, the error can be minimized between the points. Levin and Remez used four points with the abscissa of the points being the points of limiting deviation of the Chebyshev polynomial. Their method minimized the deviation at these four points. There are three points where their curve crossed the arc between the four points and had zero deviation. The use of three instead of four points to define the circle and the minimization of the error between the points enables design curves to be made for many types of functions.

The formulation of a circle by three points can be found in several books [20] and is based on the Pythagorean theorem. Thus, the circle with radius R and center, $P_o = (x_o, y_o)$, in the cartesian plane is

$$(x - x_o)^2 + (y - y_o)^2 = R^2 \tag{12}$$

The following formulation is based on a program for use on a Hewlett-Packard model 9100B calculator.

Circle Determined by Three Points

Figure 2 shows a general function, $y = f(x)$, with a circular arc imposed on it. The three points (x_1, y_1), (x_2, y_2), and (x_3, y_3) are derived from the original function $f(x)$. Using the form of equation (12) for the three points gives

$$(x_1 - x_o)^2 + (y_1 - y_o)^2 = R^2 \tag{13}$$

$$(x_2 - x_o)^2 + (y_2 - y_o)^2 = R^2 \tag{14}$$

$$(x_3 - x_o)^2 + (y_3 - y_o)^2 = R^2 \tag{15}$$

By combining expressions (13) and (14) and rearranging, the following is obtained

$$\underbrace{(x_2 - x_1)}_{A}\underbrace{(x_2 + x_1)}_{B} + \underbrace{(y_2 - y_1)}_{C}\underbrace{(y_2 + y_1)}_{D} = 2x_o(x_2 - x_1) + 2y_o(y_2 - y_1) \tag{16}$$

Substituting the variables and again rearranging obtains

$$x_o = \underbrace{\frac{AB+CD}{2A}}_{K_1} - \underbrace{\left(\frac{C}{A}\right)}_{N_1} y_o, \tag{17}$$

From equation (17), the following results

$$x_o = K_1 - N_1 y_o \tag{18}$$

If the same procedure is used for equations (13) and (15), the same general formulation is obtained

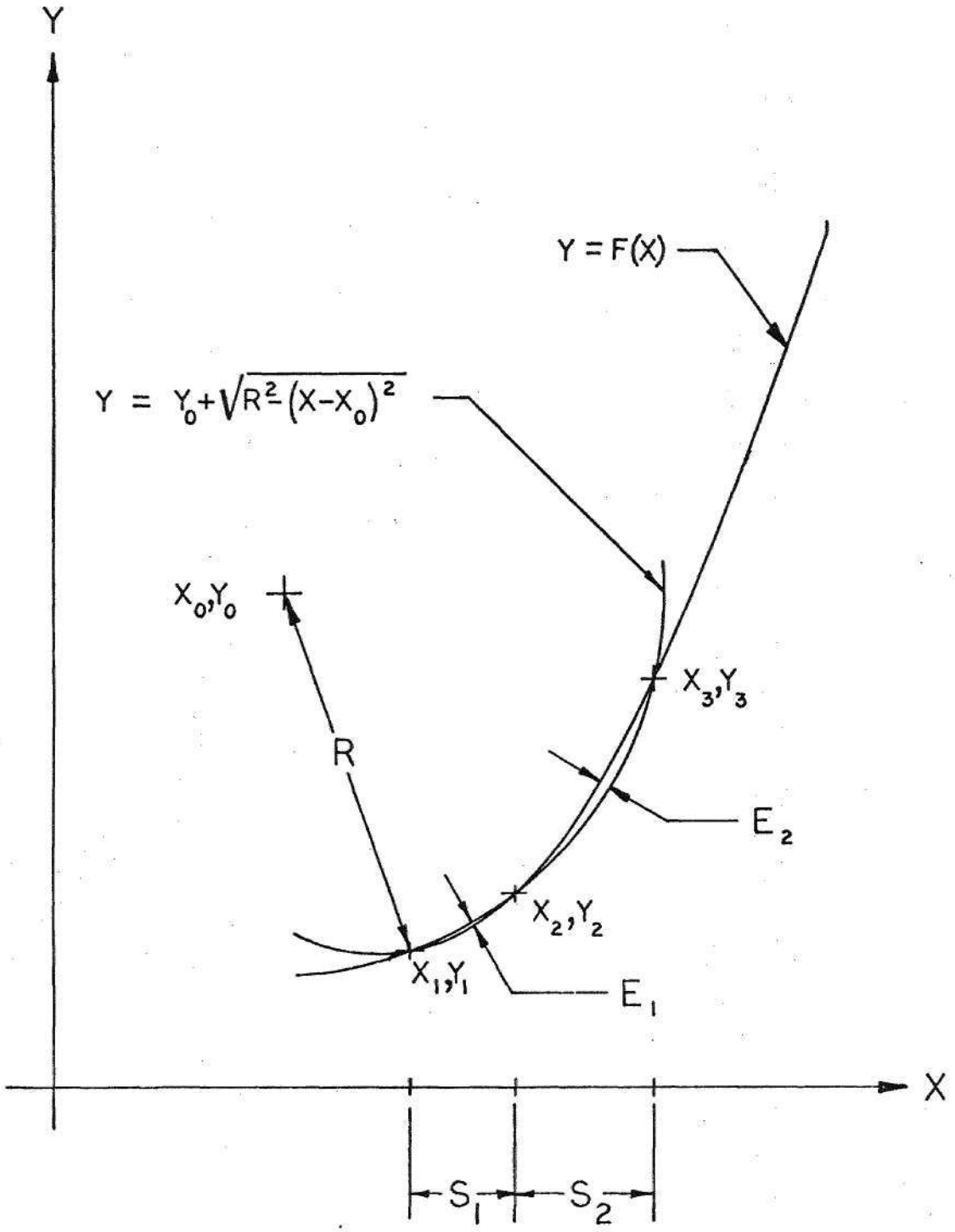


Figure 2. Circular Arc Approximation of a Function

$$\underbrace{(x_3-x_1)}_{a}\underbrace{(x_3+x_1)}_{b} + \underbrace{(y_3-y_1)}_{c}\underbrace{(y_3+y_1)}_{d} = 2x_o(x_3-x_1) + 2y_o(y_3-y_o) \qquad (19)$$

or

$$x_o = \underbrace{\frac{ab+cd}{2a}}_{K_2} - \underbrace{\left(\frac{c}{a}\right)}_{N_2} y_o \qquad (20)$$

Therefore equation (20) becomes

$$x_o = K_2 - N_2 y_o \qquad (21)$$

Combining expressions (18) and (21) obtains the following

$$y_o = \frac{K_2 - K_1}{N_2 - N_1} \qquad (22)$$

Equations (21) and (22) areused to solve for the circle's center (x_o,y_o). The radius R can be found from Equation (13).

Distance Function

Instead of using a vertical difference to determine the error curve as used in Levin's paper [3], a radial difference as mentioned by Remez [19] was thought to produce a better approach to the solution of the problem. The radial difference is the difference between the radius of the circular arc determined by the points (x_1,y_1), (x_2,y_2) and (x_3,y_3), and the distance from the original curve f(x) to the center of the circle,

(x_o, y_o). This error E is expressed by

$$E = R - \sqrt{(x-x_o)^2 + (y-y_o)^2} \tag{23}$$

For most increasing or decreasing functions, this radial difference will produce a maximum error, E_1, in one direction for the segment S_1 and a maximum error, E_2, in the other direction for the segment, S_2 (See Figure 2). The shape of this curve resembles a sine function and exhibits some of the characteristics of a Chebyshev approximation. For the three points, the curve alternates twice. The error is zero at the three designated points and reaches a minima or maxima between the points. This characteristic of the curve is somewhat like the way the structural error varies in four-bar linkage, function generators [21]. However, the structural error uses Chebshev spacing to determine the required spacing over an interval.

To determine the optimum spacing S_1 and S_2 between the points x_1, x_2, and x_3, several computer programs were written. Using a Supernova mini-computer programmed for BASIC language, most increasing functions, such as parabolas, ellipses, etc., would have about the same error E_1 and E_2 for an equal spacing between the three points x_1, x_2 and x_3. This assumption seems valid since Stone [1] found for the continuous case that the end points of his linear segments were spaced out at equal intervals over the whole interval. The use of equal spacing will allow the design curves to be drawn for many classes of functions.

Two methods were used to determine the maximum error between the points. The first is based on the Newton-Rapheson method. Taking the first derivative of the error equation (23) obtains

$$E' = - \left[\frac{(x-x_o) + y'(y-y_o)}{(x-x_o)^2 + (y-y_o)^2} \right] \tag{24}$$

Equating (24) to zero will determine where the maximum will occur. Since the denominator will vanish, only the numerator is considered

$$(x-x_o) + y'(y-y_o) = 0 \tag{25}$$

Using the Newton-Rapheson method to determine the roots of the equation which states

$$x_{n+1} = x_n - \left[\frac{h(x)}{h'(x)} \right] \tag{26}$$

Let

$$h(x) = (x-x_o) + y'(y-y_o) \tag{27}$$

$$h'(x) = 1 + y'^2 + y''(y-y_o) \tag{28}$$

Substituting back into equation (26) gives

$$x_{n+1} = x_n - \left[\frac{x_n-x_o + y_n'(y_n-y_o)}{1 + y_n'^2 + y_n''(y_n-y_o)} \right] \tag{29}$$

where $y_n = f(x_n)$, the original function. Equation (29) can be iterated to determine the point where the maximum error occurs. From this and expression (23), the error can be determined.

Since expression (29) requires the computation of the original function's first and second derivatives, a second and somewhat easier method to implement for various functions is to divide the intervals S_1 and S_2 into many sections along the x direction. From these distinct x's and Equation (23) the error is found. This error will start at zero, go to a maximum, and then return to zero at the end. The Supernova can be programmed to remember just the maximum value that occurs for the intervals.

Quadratic Functions

A general quadratic equation [20] is of the form

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0 \tag{30}$$

Solving in terms of y yields

$$y = -\frac{(Bx+E) \pm \sqrt{(Bx+E)^2 - 4C(Ax^2+Dx+E)}}{2C}, \text{ if } C \neq 0 \tag{31}$$

or

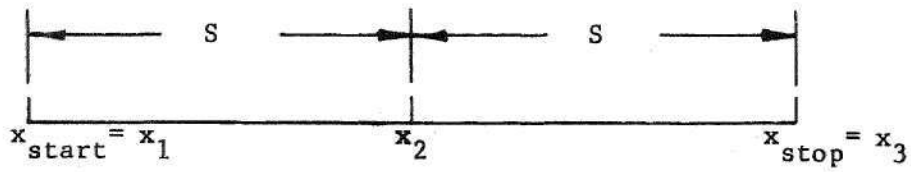
$$y = -\frac{(Ax^2 + Dx + F)}{Bx + E}, \text{ if } C = 0 \tag{32}$$

If the surface in Equation (30) is not rotated about the coordinate axis, the variable B is equal to zero and omitted. If a quadratic function is encountered, that is translated and/or rotated, no loss of generality will occur. The general quadratic program as well as the design curves to follow can be used on these translated/rotated figures if the proper coordinate transformations are used.

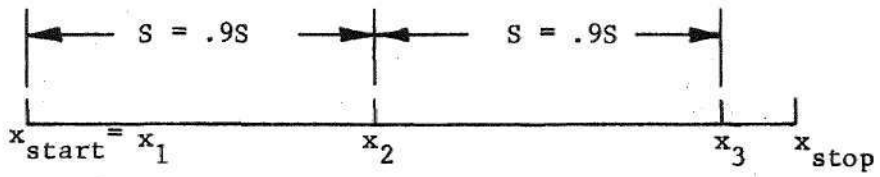
To supplement the design curves that will be presented for parabolas, ellipses, and hyperbolas, a generalized computer program was

written for Equation (30). The quadratic surface must be defined over some interval, x_{start} to x_{stop}. This general program appears in Appendix A and uses BASIC language.

Using x as the stepping variable, the program sections the distance between x_{start} and x_{stop}. The first time through the program, one arc is fitted to the entire segment. The errors E_1 and E_2 are found by using the stepping technique described in the previous section. If the error is greater than the predetermined error programmed at the beginning of the program, the size of the partition is reduced by 10%. The following Figure 3 shows the method of reducing the partition.



a. First Time



b. Next Time

Figure 3. Method of Reducing the Interval

The distance between the points can be expressed as

$$x_2 = x_1 + S \tag{33}$$

$$x_3 = x_1 + 2S \tag{34}$$

For each arc out of tolerance the process is continued and the partition is reduced until the error is within the given error. The point x_3 becomes the new beginning point x_1 for the next arc. The process of finding each arc is repeated until the end point x_{stop} is reached as the end point of the last arc.

Since this program was designed for use with numerical milling machines, provision has been made for programming the cutter diameter. The program takes the end point of the arc and the center of the arc. By computing the distance and angle between the two points, the appropriate cutter radius is either added or subtracted to give the desired offset. Templates can then be made by either specifying a plus or minus along with the cutter diameter. Figure 4 shows the relationship between the arc and the cutter.

This program along with the examples in Appendix A is intended for use in larger facilities that have the computing capabilities similar to the Supernova. The method still uses iteration to achieve the approximations. However, it is easier to utilize than the previous methods.

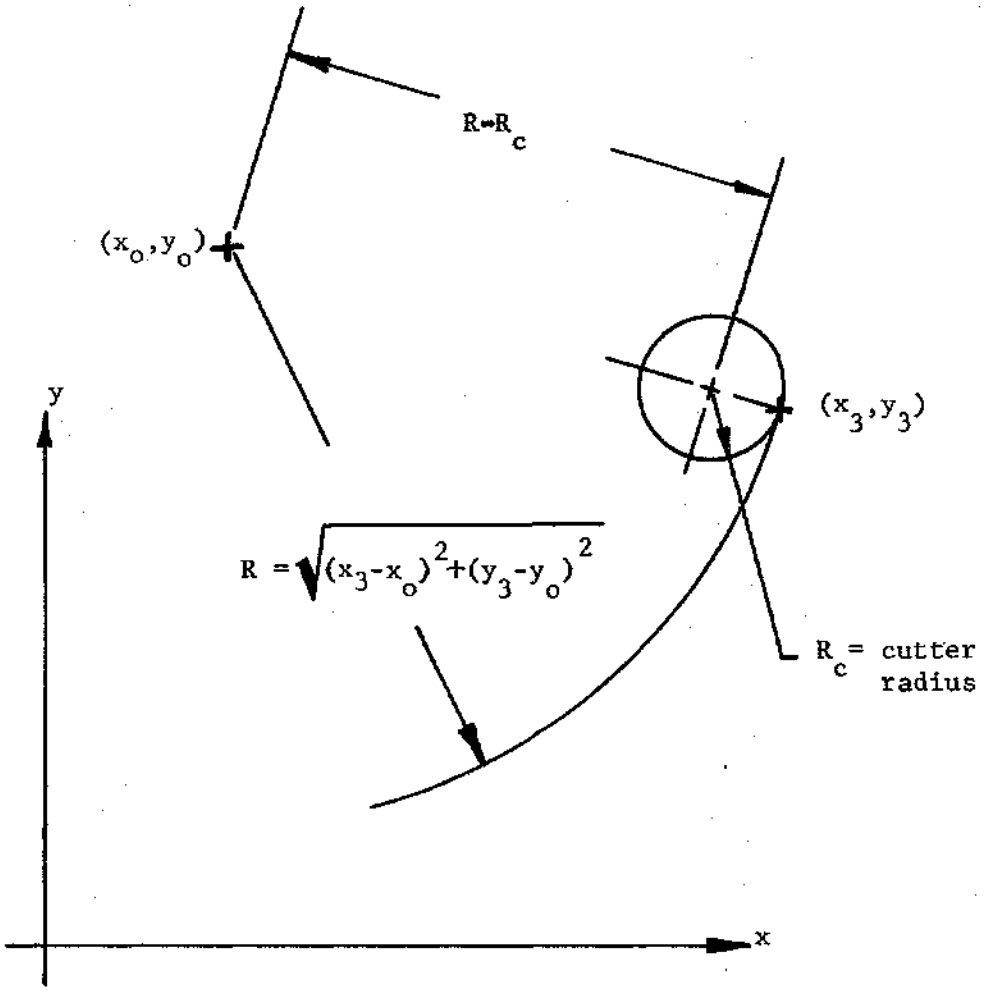


Figure 4. Cutter to Arc Relationship

CHAPTER III

APPROXIMATION OF PARABOLAS, ELLIPSES, AND HYPERBOLAS

Using the theory presented in the previous chapter, several quadratic equations were investigated. The iteration method of the general quadratic equation (Appendix A) required a small, sophisticated computer to solve for the necessary circular arcs to satisfy the given error between the arc and the function, f(x). Since most quadratic functions after rotation or translation can be described as parabolas, ellipses, or hyperbolas, these were investigated to produce design curves that would eliminate the iteration and provide a direct method of determining the number of circular arcs that would be required. These design curves would require less sophisticated and less expensive calculators, such as the Hewlett-Packard 9100B, to produce the required cartesian coordinates for the circular arcs.

These design curves represent an original contribution for the use of circular arc interpolation in numerical control machining. By using normalized parameters, these curves can be applied to many similar types of quadratic figures in several machining applications. The following sections will present the formulation and the design curves for the three quadratic functions. An example found in Appendix E demonstrates the use of these design curves.

Parabola

The parabola is perhaps the easiest of the quadratic planes to define mathematically. This type of curve is used in many fields from mirrors to microwave antenna reflectors. By letting B, C, D, and F equal zero in Equation (30) the following is obtained

$$y = \frac{Ax^2}{E} \tag{35}$$

This is commonly rewritten into the form

$$y = \frac{x^2}{4f} \tag{36}$$

where f is the focal length of the parabola.

In order to provide design curves, the various parameters must be dimensionless so that the curves will apply to broad classes of the same function. A common parameter used with paraboloids of revolution is the focal length divided by the overall diameter of the surface; this is known as the f/D (f over D) ratio.

A separate BASIC program found in Appendix B was used to generate the curves in Figure 5. The abscissa represents the number of equal segments into which the radius of the paraboloid must be divided for a specific error, diameter, and focal length. Once the radius is sectioned into equal segments and the midpoint of each segment known, these points can be used to construct the necessary arcs (see the section on Circle Determined by Three Points and the example given in Appendix E).

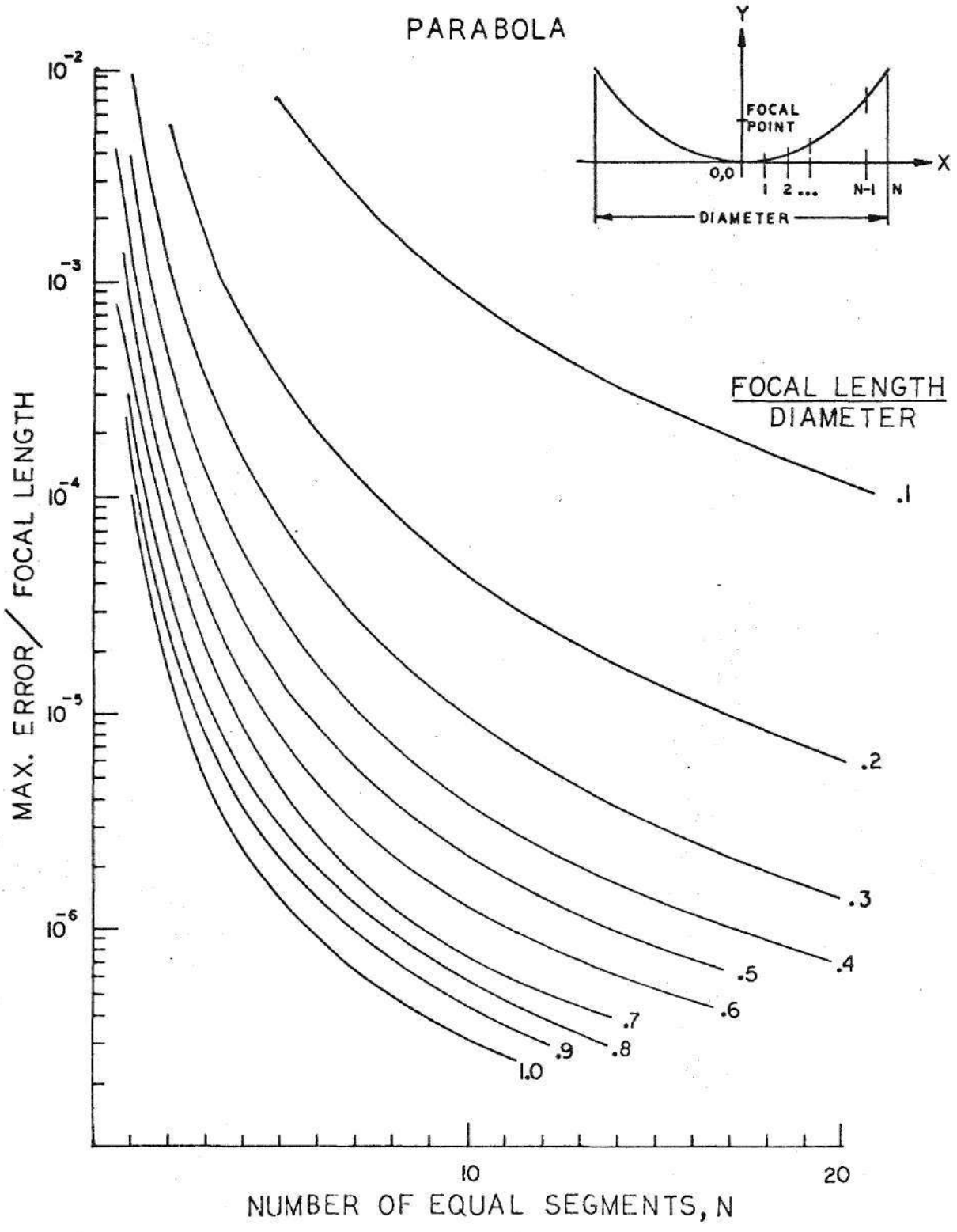


Figure 5. CAR Design Curve for Parabola

Ellipse

As with the parabola, the ellipse as well as the hyperbola will be presented in "standard" form, i.e. the plane is neither translated nor rotated and is centered on the axis of symmetry. With the exception of a sign difference, the ellipse and hyperbola share similar equations; the equation for the ellipse being

$$\frac{x^2}{A^2} + \frac{y^2}{B^2} = 1 \tag{37}$$

For the first quadrant, Equation (37) is rearranged into the form

$$y = \frac{B}{A}\sqrt{A^2 - x^2} \tag{38}$$

A and B are the major and minor radii, respectively.

The focal distance, C, of the ellipse can be determined by the following

$$C^2 = A^2 - B^2 \tag{39}$$

All three of these surfaces, the ellipse, the hyperbola, and the parabola, possess a property called eccentricity, e. This is the ratio of the distance between a point of the curve and the focus to the distance between the point and the directrix, or

$$e = \frac{C}{A} \tag{40}$$

For parabolas, the eccentricity is equal to one, less than one for ellipses, and greater than one for hyperbolas.

Being a bounded curve, the eccentricity automatically defines the major and minor radii if the focal distance is given. Thus, the eccentricity of an ellipse is similar to the f/D ratio for a parabola because each is a dimensionless parameter and defines the curve. By using the eccentricity and the dimensionless error/focal length, the design curves can be drawn.

Figure 6 was generated in a manner similar to the design curve for the parabola. The BASIC program written for the Supernova is found in Appendix C. The utilization of Figure 6 is the same as for the parabola.

Hyperbola

The hyperbola in standard form, symmetric about the y-axis, is

$$\frac{y^2}{B^2} - \frac{x^2}{A^2} = 1 \tag{41}$$

where B is the vertex of the hyperbola on the y-axis. Since only one half of the hyperbola is being considered in this research, the first quadrant equation is

$$y = \frac{B}{A}\sqrt{A^2 + x^2} \tag{42}$$

If the focal length is given, the parameter A can be determined from

$$C^2 = A^2 + B^2 \tag{43}$$

In the configuration about the y-axis, the eccentricity is

$$e = \frac{C}{B} \tag{44}$$

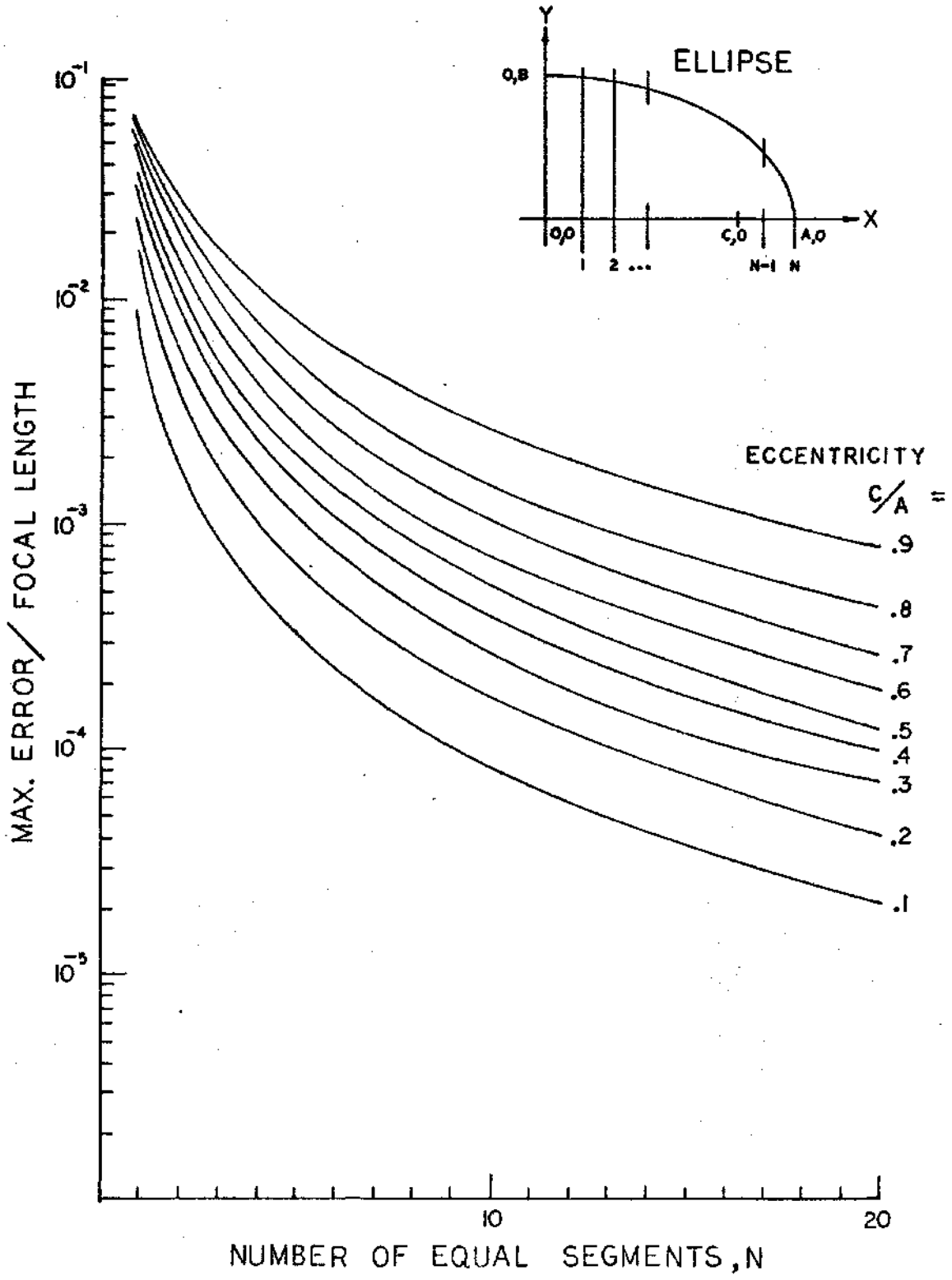


Figure 6. CAR Design Curve for Ellipse

The hyperbola exhibits the same properties of eccentricity and focal length as the ellipse. However, the hyperbola must be truncated in the same manner as the parabola to completely define the dimesionless parameters for the design curves. Therefore, both the eccentricity and the f/D ratio are needed.

Since the eccentricity and f/D ratio can go to infinity, only a few design curves shown in Figures 7 to 10 are presented. These were produced from the BASIC program in Appendix D.

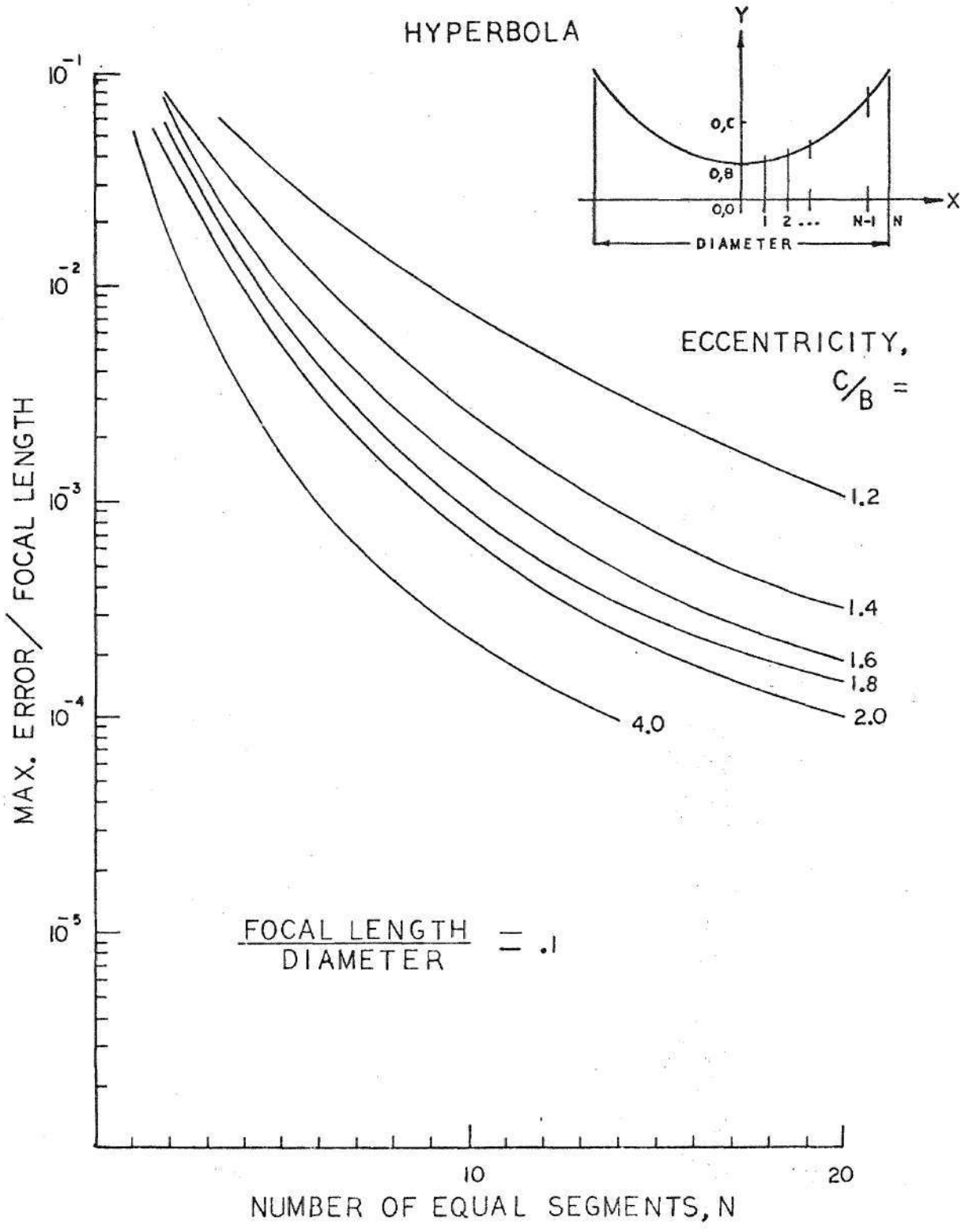


Figure 7. CAR Design Curve for Hyperbola, f/D = .1

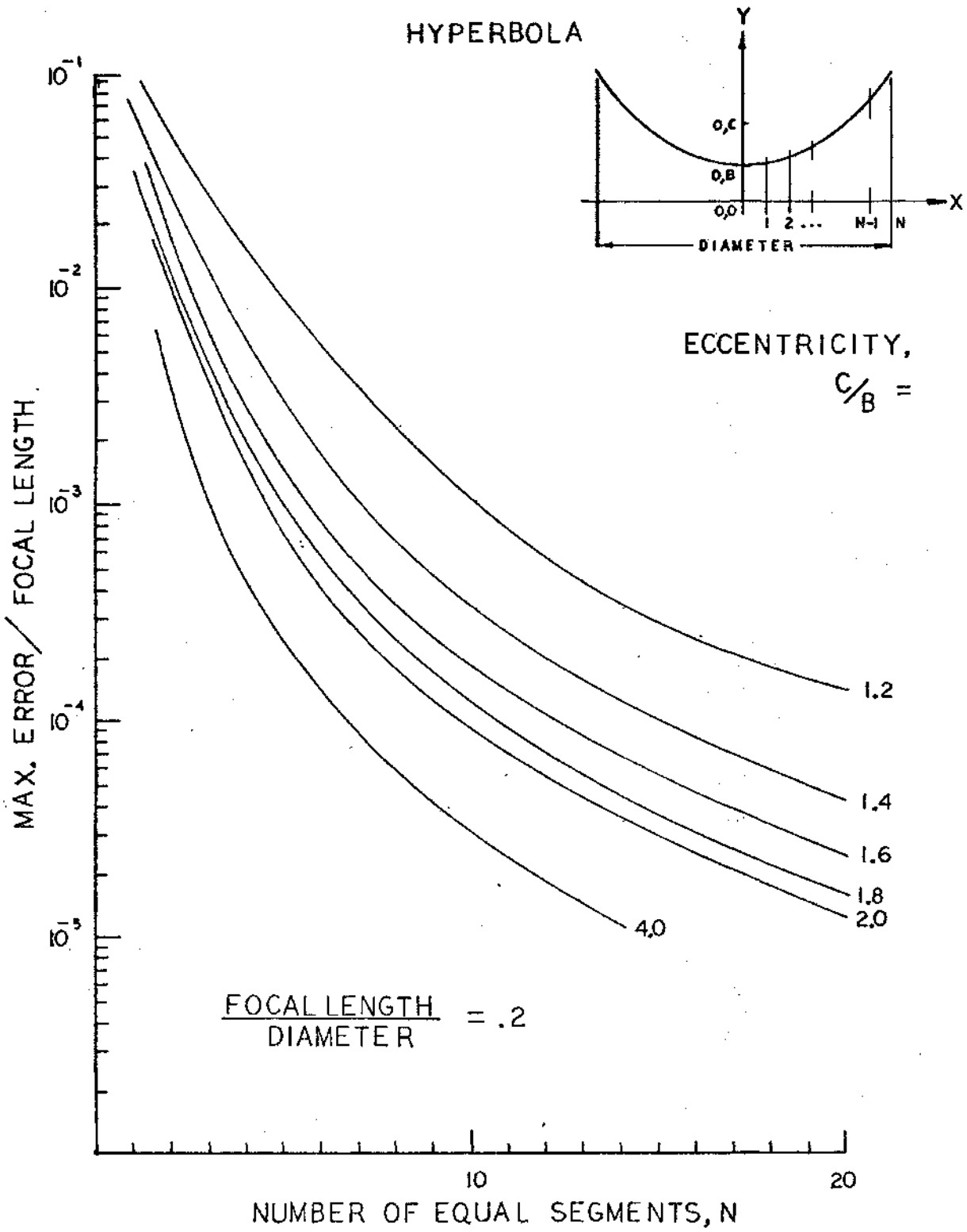


Figure 8. CAR Design Curve for Hyperbola, f/D = .2

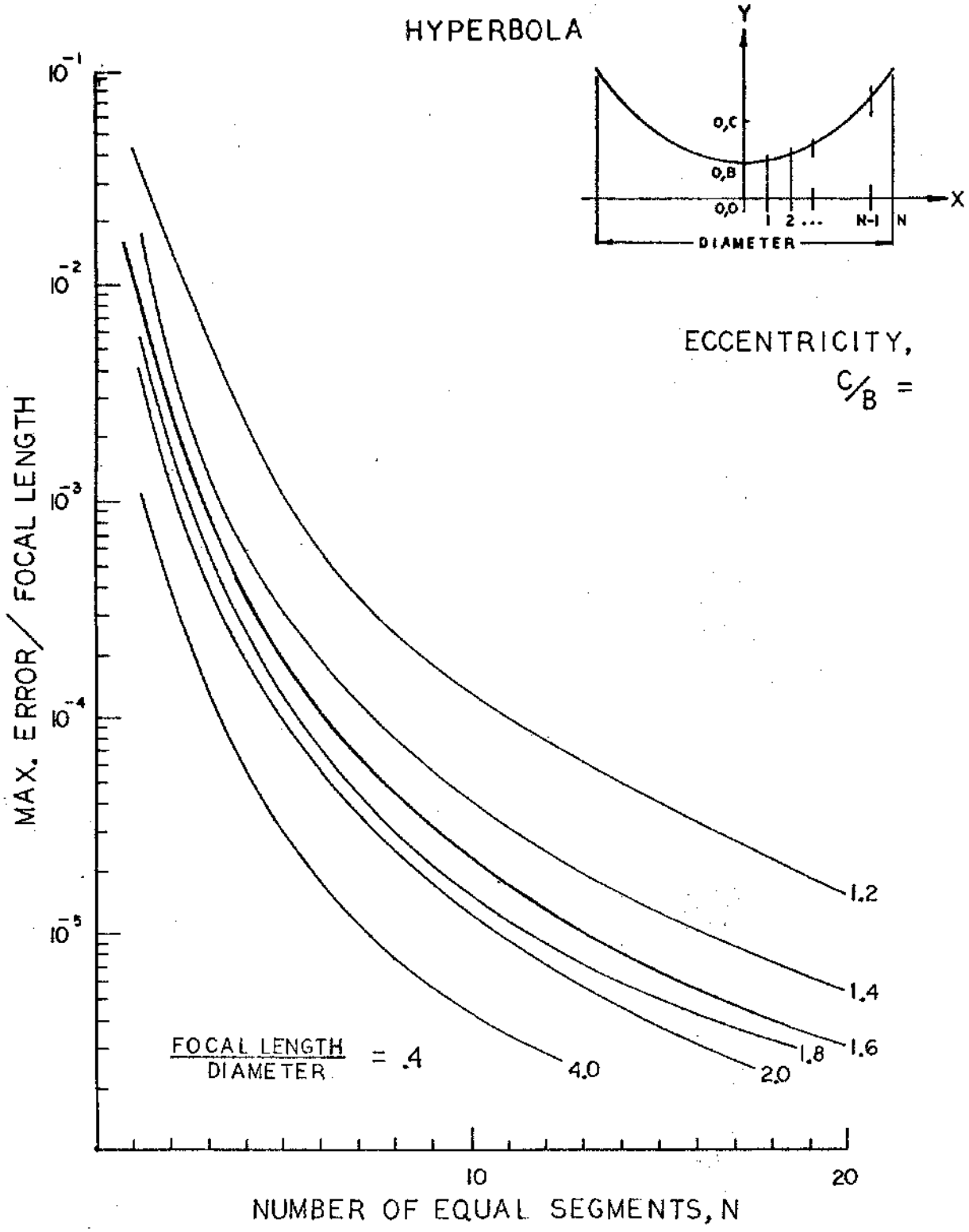


Figure 9. CAR Design Curve for Hyperbola, f/D = .4

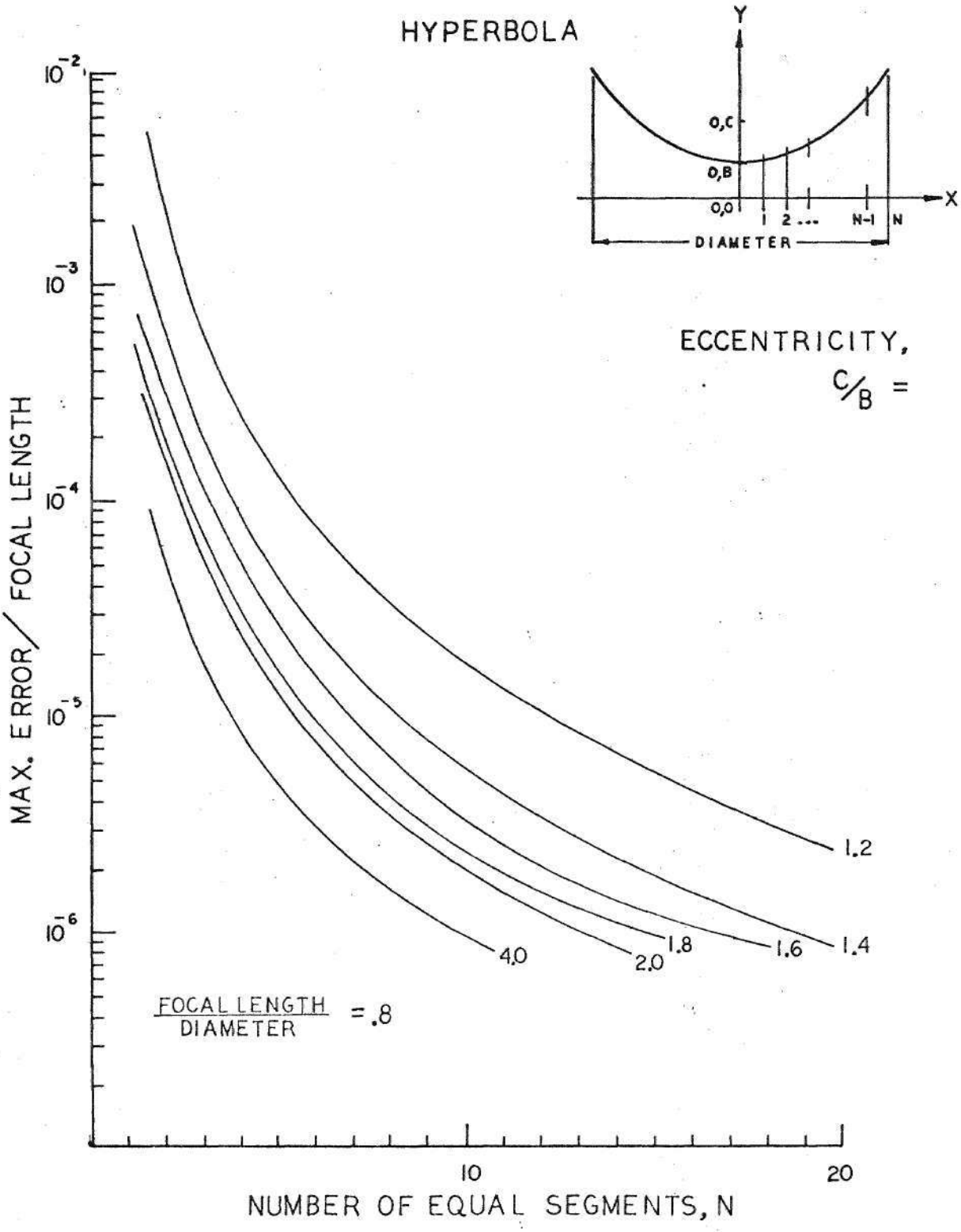


Figure 10. CAR Design Curve for Hyperbola, f/D = .8

CHAPTER IV

CONCLUSIONS AND RECOMMENDATIONS

Several conclusions were reached during the course of this research. In particular, the following were noted.

From the literature review, previous approaches to the use of circular arc approximation for numerical control were not satisfactory since the methods required either iterations or simultaneous solutions before the error between the approximated curve and the circular arc was known.

The use of three points to define a circle and the corresponding minimization of the error between the points enabled design curves to be made for many functions.

In the investigation of quadratic functions, certain parameters, such as focal length, eccentricity, and major/minor radii, formed normalized relationships so that computer programs could be written. These programs solved for the error between the two curves directly. The normalized relationships were expressed graphically and eliminated some of the shortcomings of previous methods. This original graphical method has a broad application in numerical control and other fields.

While performing this research, several areas where further work might be beneficial were noted. These recommendations should be helpful to anyone who might be working in this same field.

Trigonometric, exponential, higher-order polynomial, rational, and other functions should be investigated by using the circular arc approximation method. This would increase the number of design curves so that a single publication might contain curves for specific fields of interest.

The design curves for this research used an equal spacing along the x-axis to determine the number and size of the circular arcs. A variable spacing would give different sizes of arcs but would provide a better fit over a longer portion of the approximated curve. The sizing of the arc could use the radius of curvature or the arc length of the original curve to determine the spacing.

The slope discontinuity where one arc ends and another begins could be eliminated by making the arcs tangent to each other where they meet. An analysis of this problem would require an entirely different type of formulation of the circular arc problem.

APPENDICES

APPENDIX A

GENERAL QUADRATIC PROGRAM - BASIC LANGUAGE

```
10  DEF  FNA(X)= SQR (((B*B-4*C*A)*X*X)+((2*B*E-4*C*D)*X)+E*E-4*C*F)
15  DEF  FNB(X)=(1/(2*C))*(-(B*X)-E- FNA(X))
20  DEF  FNC(X)=(-(A*X*X)-(D*X)-F)/((B*E)+E)
22  PRINT
24  PRINT "THIS PROGRAM WILL APPROXIMATE THE QUADRATIC EQUATION"
25  PRINT "     A X↑2 + B XY + C Y↑2 + D X + E Y + F = 0"
26  PRINT "WITH CIRCULAR ARCS.  SINCE THIS PROGRAM STEPS X, THE "
27  PRINT "EQUATION MUST CONTAIN VALID VALUES OF Y FOR ALL X."
33  PRINT
34  PRINT
35  PRINT "INTER THE VARIABLES FOR A,B,C,D,E,&F."
40  PRINT "A=";
45  INPUT A
50  PRINT
55  PRINT "B=";
60  INPUT B
65  PRINT
70  PRINT "C=";
75  INPUT C
80  PRINT
85  PRINT "D=";
90  INPUT D
95  PRINT
100  PRINT "E=";
105  INPUT E
110  PRINT
115  PRINT "F=";
120  INPUT F
125  PRINT
130  PRINT "X START =";
131  INPUT U
132  PRINT
135  PRINT "X STOP =";
140  INPUT Z
145  PRINT
150  PRINT "ALLOWABLE ERROR =";
155  INPUT E1
160  PRINT
165  PRINT "CUTTER DIAMETER =";
170  INPUT V
171  PRINT
175  PRINT
176  PRINT "X","Y","I","J"
```

```
180  LET X3=U
181  LET Q=1
185  LET S=(Z-X3)/2
195  LET X1=X3
200  LET X=X1
201  IF C= 0 GOTO  1000
202  LET Y1= FNB(X)
205  LET X2=X1+S
210  LET X=X2
215  IF C= 0 GOTO  1050
220  LET Y2= FNB(X)
222  LET Y=Y2
225  GOSUB  6000
230  LET K1=K
235  LET N1=N
240  LET X3=X1+(2*S)
245  LET X=X3
250  IF C= 0 GOTO  1100
255  LET Y3= FNB(X)
257  LET Y=Y3
260  GOSUB  6000
265  LET K2=K
270  LET N2=N
275  LET Y0=(K2-K1)/(N2-N1)
280  LET X0=K2-(N2*Y0)
285  LET X=X1
290  LET Y=Y1
295  GOSUB  7000
300  LET R1=R
305  GOSUB  8000
310  IF E2<E1 GOTO  400
313  LET S=.9*S
320  GOTO  205
400  IF Q=1 GOTO  2000
405  LET X=X3
410  LET Y=Y3
500  LET M= ATN ((Y-Y0)/(X-X0))
505  LET R=R1-V/2
510  LET X=R* COS (M)
515  LET Y=R* SIN (M)
520  LET X=X+X0
525  LET Y=Y+Y0
526  IF Q=1 GOTO  2500
527  PRINT X,Y,X0,Y0
530  IF X3>=Z GOTO  9999
531  LET Q=2
535  GOTO  185
```

```
1000  LET Y1= FNC(X)
1005  GOTO  205
1050  LET Y2= FNC(X)
1055  GOTO  222
1100  LET Y3= FNC(X)
1105  GOTO  257
1150  LET Y= FNC(X)
1155  GOTO  8030
2000  LET X=X1
2005  LET Y=Y1
2010  GOTO  500
2500  PRINT X,Y
2505  LET Q=2
2510  GOTO  400
6000  LET K=((X-X1)*(X+X1)+(Y-Y1)*(Y+Y1))/(2*(X-X1))
6005  LET N=((Y-Y1)/(X-X1))
6010  RETURN
7000  LET R= SQR ((X-X0)*(X-X0)+(Y-Y0)*(Y-Y0))
7005  RETURN
8000  LET S1=(X3-X1)/20
8005  LET E2= 0
8010  LET X=X1
8015  FOR I= 0 TO 20
8020    LET X=X1+S1*I
8025    IF C= 0 GOTO  1150
8026    LET Y= FNB(X)
8030    GOSUB  7000
8035    LET E3=R1-R
8040    LET W= SGN (E3)
8045    LET E3= ABS (E3)
8050    IF E3<E2 GOTO  8060
8055    LET E2=E3
8060  NEXT I
8065  RETURN
9999  END
```

SAMPLE RUN OF GENERAL QUADRATIC EQUATION PROGRAM

```
THIS PROGRAM WILL APPROXIMATE THE QUADRATIC EQUATION
    A X↑2 + B XY + C Y↑2 + D X + E Y + F = 0
WITH CIRCULAR ARCS.  SINCE THIS PROGRAM STEPS X, THE
EQUATION MUST CONTAIN VALID VALUES OF Y FOR ALL X.


INTER THE VARIABLES FOR A,B,C,D,E,&F.
A=? 1
B=? 0
C=? .325
D=? 0
E=? -202
F=? 0
X START =? 0
X STOP =?  60
ALLOWABLE ERROR =? .0001
CUTTER DIAMETER =? 0

X              Y              I              J
 0              0
 9.00566        .401733       -5.66769E-3     101.251
 15.8942        1.25317       -.128294        102.597
 22.5142        2.51967       -.465679        104.707
 27.578         3.78831       -1.02804        107.232
 32.985         5.43391       -1.80574        110.045
 37.9909        7.22939       -2.92536        113.431
 43.0259        9.30399       -4.38739        117.227
 47.8199        11.5348       -6.19414        121.344
 52.5387        13.9796       -8.31271        125.656
 56.9445        16.4906       -10.781         130.2
 60             18.3646       -13.3755        134.574
```

APPENDIX B

BASIC PROGRAM FOR PARABOLA DESIGN CURVES

```
15  FOR F=2.4 TO 24 STEP 2.4
16    PRINT
17    PRINT
18    PRINT
20    LET D=24
35    LET R0=D/2
36    PRINT "F/D=";F/D
37    PRINT
38    PRINT "N","E/F"
39    PRINT
40    FOR T=1 TO 20
44      PRINT T,
45      LET S=R0/(2*T)
50      LET U=0
51      LET E1=0
55      FOR B=1 TO T
60        IF B=1 GOTO  4000
70        LET X1=X3
75        LET Y1=(X3*X3)/(4*F)
80        GOSUB  5000
85        LET X2=X
90        LET Y2=Y
95        GOSUB  6000
100        LET K1=K
105        LET N1=N
110        GOSUB  5000
115        LET X3=X
120        LET Y3=Y
121        GOSUB  6000
122        LET K2=K
135        LET N2=N
140        LET Y0=((K2-K1)/(N2-N1))
145        LET X0=(K2-(N2*Y0))
150        LET X=X1
155        LET Y=Y1
160        GOSUB  7000
165        LET R1=R
170        LET M=(Y2-Y1)/(X2-X1)
185        GOSUB  8000
190        LET A=E
200        LET M=(Y3-Y2)/(X3-X2)
```

```
210          GOSUB  8000
245          IF A>E GOTO  3000
247          IF E>E1 GOTO  3500
250        NEXT B
252        PRINT E1/F
255      NEXT T
256    NEXT F
260    GOTO  9000
3000    LET E=A
3005    GOTO  247
3500    LET E1=E
3505    GOTO  250
4000    LET X3= 0
4010    GOTO  70
5000    LET U=U+S
5001    LET X=U
5005    LET Y=(X*X)/(4*F)
5010    RETURN
6000    LET K=((X-X1)*(X+X1)+(Y-Y1)*(Y+Y1))/(2*(X-X1))
6005    LET N=((Y-Y1)/(X-X1))
6010    RETURN
7000    LET R= SQR ((X-X0)*(X-X0)+(Y-Y0)*(Y-Y0))
7005    RETURN
8000    LET X=2*F*M
8005    LET Y=F*M*M
8030    GOSUB  7000
8035    LET R2=R
8040    LET E=R2-R1
8045    LET W= SGN (E)
8050    LET E= ABS (E)
8070    RETURN
9000    END
```

APPENDIX C

BASIC PROGRAM FOR ELLIPSE DESIGN CURVES

```
10  DEF  FNA(X)=(B/A)* SQR ((A*A)-(X*X))
15  FOR U=.1 TO .9 STEP .1
16    PRINT
17    PRINT
20    LET C=1
25    LET A=C/U
30    LET B= SQR ((A*A)-(C*C))
35    PRINT "C/A=",C/A
40    PRINT
45    PRINT "N","E/C"
50    PRINT
55    FOR T=2 TO 20 STEP 2
60      PRINT T,
65      LET S=A/(2*T)
70      LET X3= 0
71      LET E1= 0
75      FOR Q=1 TO T
80        LET X1=X3
85        LET Y1= FNA(X1)
90        LET X2=X1+S
95        LET Y2= FNA(X2)
96        LET X=X2
97        LET Y=Y2
100        GOSUB  6000
105        LET K1=K
110        LET N1=N
115        LET X3=X1+(2*S)
120        LET Y3= FNA(X3)
121        LET X=X3
122        LET Y=Y3
125        GOSUB  6000
130        LET K2=K
135        LET N2=N
140        LET Y0=(K2-K1)/(N2-N1)
145        LET X0=K2-(N2*Y0)
150        LET X=X1
155        LET Y=Y1
160        GOSUB  7000
165        LET R1=R
170        GOSUB  8000
171        IF E2<E1 GOTO  175
172        LET E1=E2
175      NEXT Q
```

```
180       PRINT E1/C
185     NEXT T
190   NEXT U
191   GOTO  9999
6000LIST
6000   LET K=((X-X1)*(X+X1)+(Y-Y1)*(Y+Y1))/(2*(X-X1))
6005   LET N=((Y-Y1)/(X-X1))
6010   RETURN
7000   LET R= SQR ((X-X0)*(X-X0)+(Y-Y0)*(Y-Y0))
7005   RETURN
8000   LET S1=(X3-X1)/10
8005   LET E2= 0
8010   LET X=X1
8015   FOR I= 0 TO 10
8020     LET X=X1+S1*1
8026     LET Y= FNA(X)
8030     GOSUB  7000
8035     LET E3=R1-R
8040     LET W= SGN (E3)
8045     LET E3= ABS (E3)
8050     IF E3<E2 GOTO  8060
8055     LET E2=E3
8060   NEXT I
8065   RETURN
9999   END
```

APPENDIX D

BASIC PROGRAM FOR HYPERBOLA DESIGN CURVES

```
10  DEF  FNA(X)=(B/A)* SQR ((A*A)+(X*X))
11  LET D=24
12  LET Z=D/2
13  PRINT
14  PRINT
15  FOR C=2.4 TO 24 STEP 2.4
16    PRINT "F/D=";C/D
17    PRINT
20    FOR U=1.2 TO 2 STEP .2
25      LET B=C/U
26      PRINT "C/B=";C/B
27      PRINT
30      LET A= SQR ((C*C)-(B*B))
40      PRINT
45      PRINT "N","E/C"
50      PRINT
55      FOR T=2 TO 20 STEP 2
60        PRINT T,
65        LET S=Z/(2*T)
70        LET X3= 0
71        LET E1= 0
75        FOR Q=1 TO T
80          LET X1=X3
85          LET Y1= FNA(X1)
90          LET X2=X1+S
95          LET Y2= FNA(X2)
96          LET X=X2
97          LET Y=Y2
100          GOSUB  6000
105          LET K1=K
110          LET N1=N
115          LET X3=X1+(2*S)
120          LET Y3= FNA(X3)
121          LET X=X3
122          LET Y=Y3
125          GOSUB  6000
130          LET K2=K
135          LET N2=N
140          LET Y0=(K2-K1)/(N2-N1)
145          LET X0=K2-(N2*Y0)
150          LET X=X1
155          LET Y=Y1
160          GOSUB  7000
```

```
165           LET R1=R
170           GOSUB  8000
171           IF E2<E1 GOTO  175
172           LET E1=E2
175         NEXT Q
180         PRINT E1/C
185       NEXT T
186       PRINT
190     NEXT U
191     PRINT
195   NEXT C
200   GOTO  9999
6000LIST
6000   LET K=((X-X1)*(X+X1)+(Y-Y1)*(Y+Y1))/(2*(X-X1))
6005   LET N=((Y-Y1)/(X-X1))
6010   RETURN
7000   LET R= SQR ((X-X0)*(X-X0)+(Y-Y0)*(Y-Y0))
7005   RETURN
8000   LET S1=(X3-X1)/10
8005   LET E2= 0
8010   LET X=X1
8015   FOR I= 0 TO 10
8020     LET X=X1+S1*I
8026     LET Y= FNA(X)
8030     GOSUB  7000
8035     LET E3=R1-R
8040     LET W= SGN (E3)
8045     LET E3= ABS (E3)
8050     IF E3<E2 GOTO  8060
8055     LET E2=E3
8060   NEXT I
8065   RETURN
9999   STOP
```

APPENDIX E

EXAMPLE OF THE USE OF THE DESIGN CURVES

Having presented the design curves, a specific example to illustrate the use of these curves will be presented.

A Parabola with a 4 inch focal length and a 10 inch diameter is to be approximated with circular arcs to within 0.001 inch error between the curves.

Formulation of design curve parameters:

1. $f/D = .4$
2. Max. Error/Focal Length = 2.5×10^{-4}

From Figure 5, the required number of equal segments is 4 (using the largest whole number). To have four arcs where each is determined from three points and the end of one arc is the beginning of the next, there must be 8 equally spaced divisions between the beginning, $X_o = 0$, and the end, $X_8 = 5$. The first arc would contain X_o, X_1, X_2; the second arc, X_2, X_3, X_4; the third arc, X_4, X_5, X_6; and the fourth, X_6, X_7, X_8. The corresponding Y-values are derived from the Equation (36) for the parabola.

Once the X,Y coordinates for the nine points are computed, these can be used to find the corresponding center of each arc from the section, Circle Determined by Three Points. A small calculator, such as a Hewlett-Packard 9100B, can be programmed to do this operation. The following table summarizes the calculations:

	X	Y	Arc X_o	Center Y_o	
0	0	0			
1	.625	.0244	.0022	8.0737	(Arc 1)
2	1.250	.0977			
3	1.875	.2197	.0967	8.6588	(Arc 2)
4	2.500	.3906			
5	3.125	.6104	.4817	9.8676	(Arc 3)
6	3.750	.8789			
7	4.375	1.1963	1.3051	11.6071	(Arc 4)
8	5.000	1.5625			

Thus, the first arc begins at (0,0) and ends at (1.250, .0977) and has its center at (.0022, 8.0737). Each additional arc is computed in a similar manner. These arcs form the mean-line of the parabola; any additional corrections for cutter size must be made after this.

REFERENCES

1. H. Stone, "Approximation of Curves by Line Segments", Mathematics of Computation, 15, 1961.

2. L. B. Smith, "Drawing Ellipses, Hyperbolas, or Parabolas with a Fixed Number of Points and Maximum Inscribed Area", The Computer Journal, Vol. 14, No. 1, 1970.

3. S. S. Levin, "Design Calculation for Profiled Cutting Tools Using Circular Arcs", Machines and Tooling (English translation), Vol. 32, No. 4, 1961.

4. _____, Numerical Control Systems, Friden Division, Singer Company, Rochester, New York, 1969.

5. P. Bezier, Numerical Control, Mathematics and Application, John Wiley and Sons, London, England, 1970.

6. W. C. Zehnpfennig, "Automatic Positioning by NC", Machine Design, May 14, 1970.

7. I. W. Simon, The Numerical Control of Machine Tools, Edward Arnold Ltd., London, England, 1973.

8. J. E. Hulet,"Numerical Control, Machining by the Numbers", Design News, November 22, 1971.

9. F. J. Lavoie, "Computerized NC to Match Your Budget", Machine Design, December 14, 1972.

10. _____, APT Part Programming, IIT Research Institute, McGraw-Hill Book Company, New York, New York, 1967.

11. N. Akgerman and T. Altan, "NC's Growing Impact on Design", Machine Design, July 27, 1972.

12. R. Khol, "The Search for the Sculptured Surface", Machine Design, March 22, 1973.

13. R. V. Churchill, Fourier Series and Boundary Value Problems, McGraw-Hill Book Company, New York, New York, 1963.

14. J. R. Rice, The Approximation of Functions, Vol. 1, Addison-Wesley, Reading, Massachusetts, 1964.

15. R. Bellman, "On the Approximation of Curves by Line Segments Using Dynamic Programming", Communications of the ACM, No. 4, 1961.

16. R. Bellman, "Dynamic Programming, System Identification, and Suboptimization", J. SIAM Control, Vol. 4, No. 1, 1966.

17. S. S. L. Chang, "Adaptive Curve Fitting and Suboptimization", IEEE Transactions on Automatic Control, December, 1968.

18. B. Nicoletti and L. Mariani, "A Computer Algorithm for Optimal Curve Fitting", IEEE Transactions on Automatic Control, February, 1971.

19. E. Y. Remez and S. S. Levin, "On the Problem of Uniformly Approximate Replacement of a Curved Arc by Circular Arcs , Ukrainian Mathematical Journal, Vol. 22, No. 2, 1970.

20. J. A. Hummel, Vector Geometry, Addison-Wesley Publishing Company, Reading, Massachusetts, 1965.

21. R. S. Hartenberg and J. Denavit, Kinematic Synthesis of Linkages, McGraw-Hill Book Company, New York, New York, 1964.